***merge sort:***

def mergeSort(arr):

if len(arr) >1:

mid = len(arr)//2

L = arr[:mid]

R = arr[mid:]

mergeSort(L)

mergeSort(R)

i = j = k = 0

while i < len(L) and j < len(R):

if L[i] < R[j]:

arr[k] = L[i]

i+=1

else:

arr[k] = R[j]

j+=1

k+=1

while i < len(L):

arr[k] = L[i]

i+=1

k+=1

while j < len(R):

arr[k] = R[j]

j+=1

k+=1

def printList(arr):

for i in range(len(arr)):

print(arr[i],end=" ")

print()

if \_\_name\_\_ == '\_\_main\_\_':

arr = [int(x)for x in input().split()]

print ("Given array is", end="\n")

printList(arr)

mergeSort(arr)

print("Sorted array is: ", end="\n")

printList(arr)

***bucket sort:***

def bucket\_sort(alist):

largest = max(alist)

length = len(alist)

size = largest/length

buckets = [[] for \_ in range(length)]

for i in range(length):

j = int(alist[i]/size)

if j != length:

buckets[j].append(alist[i])

else:

buckets[length - 1].append(alist[i])

for i in range(length):

insertion\_sort(buckets[i])

result = []

for i in range(length):

result = result + buckets[i]

return result

def insertion\_sort(alist):

for i in range(1, len(alist)):

temp = alist[i]

j = i - 1

while (j >= 0 and temp < alist[j]):

alist[j + 1] = alist[j]

j = j - 1

alist[j + 1] = temp

alist = input('Enter the list of (nonnegative) numbers: ').split()

alist = [int(x) for x in alist]

sorted\_list = bucket\_sort(alist)

print('Sorted list: ', end='')

print(sorted\_list)

***Insertion sort:***

def insertionSort(arr):

for i in range(1, len(arr)):

key = arr[i]

j = i-1

while j >= 0 and key < arr[j] :

arr[j + 1] = arr[j]

j -= 1

arr[j + 1] = key

arr = [int(x)for x in input().split()]

insertionSort(arr)

for i in range(len(arr)):

print (arr[i], end=' ')

***Counting inversions:***

count=0

def msort(a,l,h):

if l<h:

mid=(l+h)//2

msort(a,l,mid)

msort(a,mid+1,h)

merge(a,l,mid,h)

def merge(a,l,m,h):

global count

c=[]

i=l

j=m+1

while i<=m and j<=h:

if a[j]>a[i]:

c.append(a[i])

i=i+1

else:

c.append(a[j])

j=j+1

count=count+len(a[i:m+1])

if i>m:

while j<=h:

c.append(a[j])

j=j+1

if j>h:

while i<=m:

c.append(a[i])

i=i+1

a[l:h+1]=c

print("Enter the array: ")

l=[int(x) for x in input().split()]

msort(l,0,len(l)-1)

print(l)

print("Number of inversions: ",count)

**bfs:**

n=int(input("Enter number of nodes:"))v={}bfs=[]val={}k=0for i in range(n): m=input("Enter vertex:" ) val[m]=k k=k+1 print("Enter adjacent nodes: ") l=list(input().split()) v[m]=ls=input("Enter source node: ")q=[]dis=[0 for i in range(n)]q.insert(0,s)dis[val[s]]=1while q!=[]: m=q.pop() bfs.append(m) for i in v[m]: if dis[val[i]]==0: dis[val[i]]=1 q.insert(0,i)print(bfs)

**DFS:**

n=int(input("Enter number of nodes:"))

v={}

dfs=[]

val={}

k=0

for i in range(n):

m=input("Enter vertex:" )

val[m]=k

k=k+1

print("Enter adjacent nodes: ")

l=list(input().split())

v[m]=l

s=input("Enter source node: ")

st=[]

dis=[0 for i in range(n)]

st.append(s)

while st!=[]:

m=st.pop()

dfs.append(m)

if dis[val[m]]==0:

dis[val[m]]=1

for i in v[m]:

if dis[val[i]]==0:

if i not in st:

st.append(i)

print(dfs)

**Stable marriage:**

mp={}wp={}n=int(input("Enter the number of men and women: "))for i in range(n): name=input("Enter the name of man: ") print("Enter his preference order: ") l=list(input().split()) mp[name]=lfor i in range(n): name=input("Enter the name of woman: ") print("Enter her preference order: ") l=list(input().split()) wp[name]=lmfree=[]wm={}def gret(a,b,c): l=wp[c] for i in l: if i==a: return(True) elif i==b: return(False)for i in mp.keys(): mfree.append(i)while mfree!=[]: m=mfree[0] w=mp[m][0] if w not in wm: wm[w]=m mp[m]=mp[m][1:] mfree=mfree[1:] else: if gret(m,wm[w],w): mfree=mfree[1:] mfree.insert(0,wm[w]) wm[w]=m mp[m]=mp[m][1:] else: mp[m]=mp[m][1:]for i in wm: print(wm[i],i)